

"Variable Redundancy Product Codes"

L. J. Weng and G. H. Sollman

Northeastern University  
Boston, Massachusetts

ABSTRACT

(This paper is intended for the Communication Technology Group, (1) Session 4: Space Communications, (2) Session 9: Data Communications - The 1967 View, and (3) Session 8: Advances in Data Communications, Telegraph Systems, and Facsimile.)

The tensor product codes, as either iterated codes or error-locating codes, possess the property of variable redundancy (allowing up to eight different levels of error control). It is shown that the implementation of an encoder for such codes can be decomposed into the implementation of the component codes. The selection of eight encoder modes yields four iterated codes and four error-locating codes that lend themselves to both adaptive coding and feedback transmission strategies.

FACILITY FORM 602	N 68-27618	
	(ACCESSION NUMBER)	(THRU)
	14	1
	(PAGES)	(CODE)
	CR-78334	07
	(NASA CR OR TMX OR AD NUMBER)	(CATEGORY)

GPO PRICE \$ \_\_\_\_\_

CFSTI PRICE(S) \$ \_\_\_\_\_

Hard copy (HC)

Microfiche (MF)

## "Variable Redundancy Product Codes"\*

L. J. Weng and G. H. Sollman

Northeastern University  
Boston, Massachusetts

### SYNOPSIS


The concept of iterated codes has been under discussion for more than a decade.<sup>1</sup> It was pointed out by Slepian<sup>2</sup> that the generator matrix of an iterated code can be expressed as the tensor product of the (two) generator matrices of its component codes. Recently the idea of an error-locating code<sup>3</sup> has been developed and its parity-check matrix was found to be the tensor product of the parity-check matrices of its component codes.<sup>4,5,6</sup> At the same time, Tang<sup>7</sup> has proposed a single encoder which may be used to encode dual cyclic codes (thereby providing two encoder modes of different redundancy). These ideas have been drawn upon to develop an eight mode encoder whose code words are of constant word block length and offer eight levels of redundancy. This coding technique is particularly suitable for channels with varying characteristics due to such factors as weather conditions, interference from adjacent channels, varying transmission distance (especially in the case of space communications), etc.

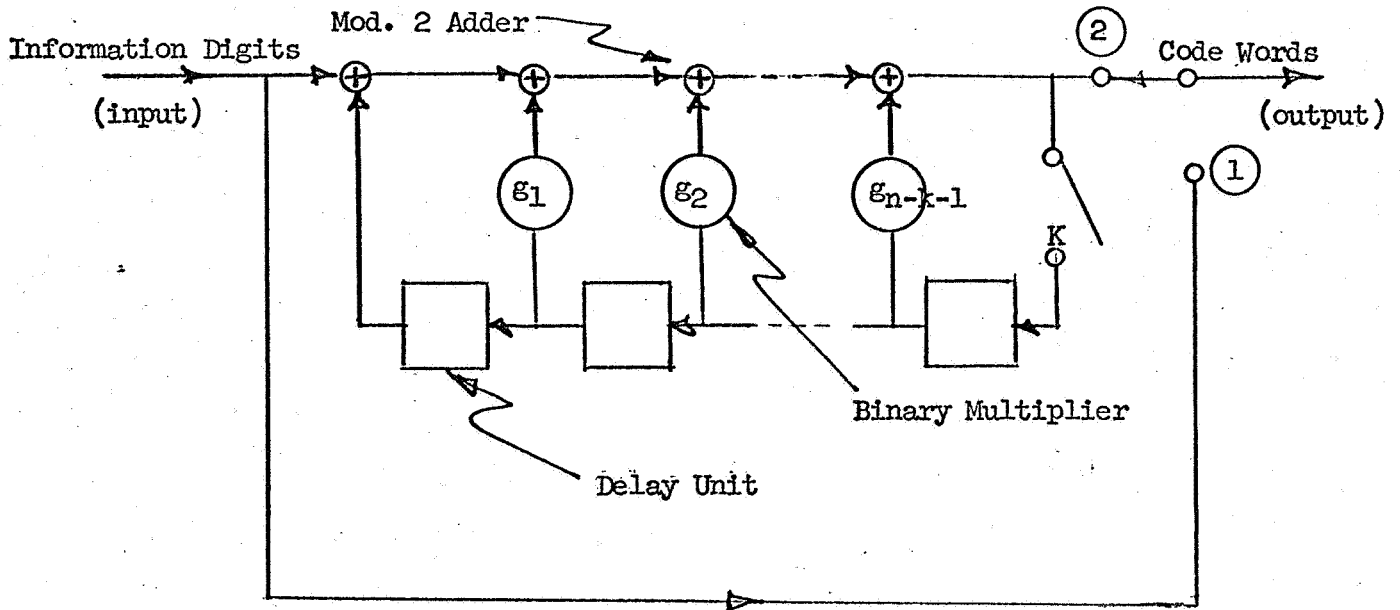
The purpose of this paper is (1) to decompose the implementation of the product code into the implementation of its component codes, and (2) to present a single encoder which will encode information according to any of eight dual product codes, and thereby yielding a wide range of error-control capabilities provided the component codes are cyclic.

It has been shown<sup>8</sup> that a systematic  $(n,k)$  cyclic code with generator polynomial  $g(x) = 1 + g_1x + g_2x^2 + \dots + g_{n-k-1}x^{n-k-1} + x^{n-k}$  and recursive polynomial  $h(x) = 1 + h_1x + h_2x^2 + \dots + h_{k-1}x^{k-1} + x^k$ , where the coefficients  $g_i$  and  $h_i$  are either 1 or 0, can be generated by either of the encoders given in Fig. 1 or Fig. 2.

---

\*This work was performed under the joint support of Air Force Cambridge Research Laboratories under Contract No. AF19(628)-3312 and ERC, NASA Grant No. NGR-22-011-013.

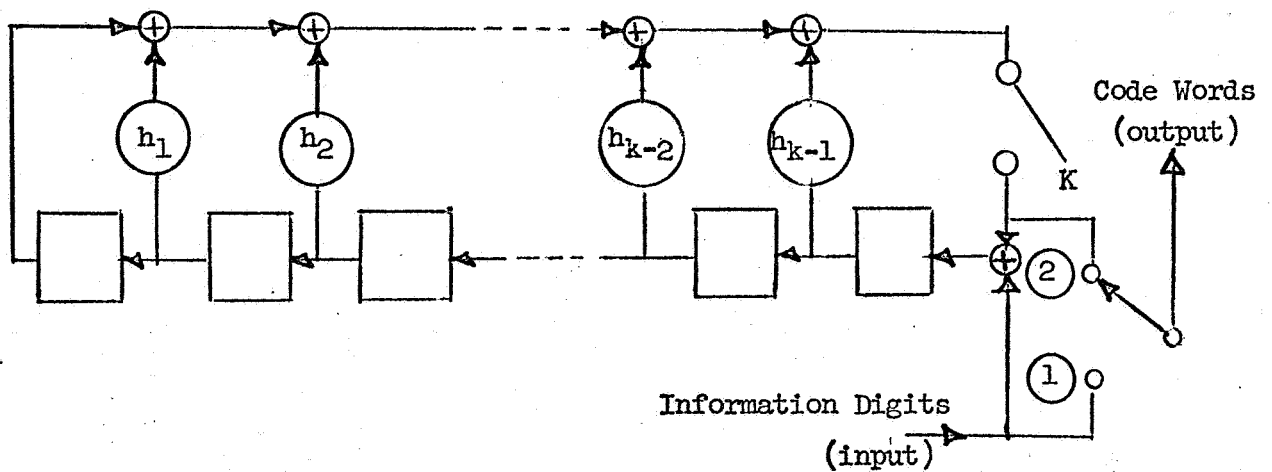




(NOTE: In binary the multiplier is either 0 or 1, with the ZERO denoting no connection and ONE denoting a completed connection.)

S is at position 1 when information digits are incoming, then switches to position 2 to receive checking digits. Switch K is closed only when S is in position 2.

Fig. 1 Encoding  $(n,k)$  Cyclic Code by Polynomial  $g(x)$



Switch S is at position 1 when information digits are incoming, then switches to position 2 for the checking digits. Switch K is closed only when S is in position 1.

Fig. 2 Encoding  $(n,k)$  Cyclic Code by Polynomial  $h(x)$

With the exception of the input output connections and switch timing, the encoders of Fig. 1 and 2 are seen to be identical. It is this fact which is exploited to encode dual cyclic codes by a single encoder.<sup>7</sup>

### Iterated Codes

There are several ways to implement the iterated codes and error-locating codes.<sup>8</sup> Here we illustrate only the encoding circuits for the product codes which may be easily converted from one code to the other. The fundamental concept of the encoder is to consider the tensor product as gating the second component code by the first one.

Let the generator matrix  $G^9$  of an iterated code be the product of two component matrices,  $G_1$  and  $G_2$ , of  $(n_1, k_1)$  and  $(n_2, k_2)$  codes, respectively:

$$G = G_1 \otimes G_2 = \begin{bmatrix} a_{11}^{(1)} & a_{12}^{(1)} & \dots & a_{1n_1}^{(1)} \\ a_{21}^{(1)} & a_{22}^{(1)} & \dots & a_{2n_1}^{(1)} \\ \vdots & \vdots & & \vdots \\ a_{k_1 1}^{(1)} & a_{k_1 2}^{(1)} & \dots & a_{k_1 n_1}^{(1)} \end{bmatrix} \otimes \begin{bmatrix} a_{11}^{(2)} & a_{12}^{(2)} & \dots & a_{1n_2}^{(2)} \\ a_{21}^{(2)} & a_{22}^{(2)} & \dots & a_{2n_2}^{(2)} \\ \vdots & \vdots & & \vdots \\ a_{k_2 1}^{(2)} & a_{k_2 2}^{(2)} & \dots & a_{k_2 n_2}^{(2)} \end{bmatrix} \quad (1)$$

Each row vector of the matrix  $G$  is of the form

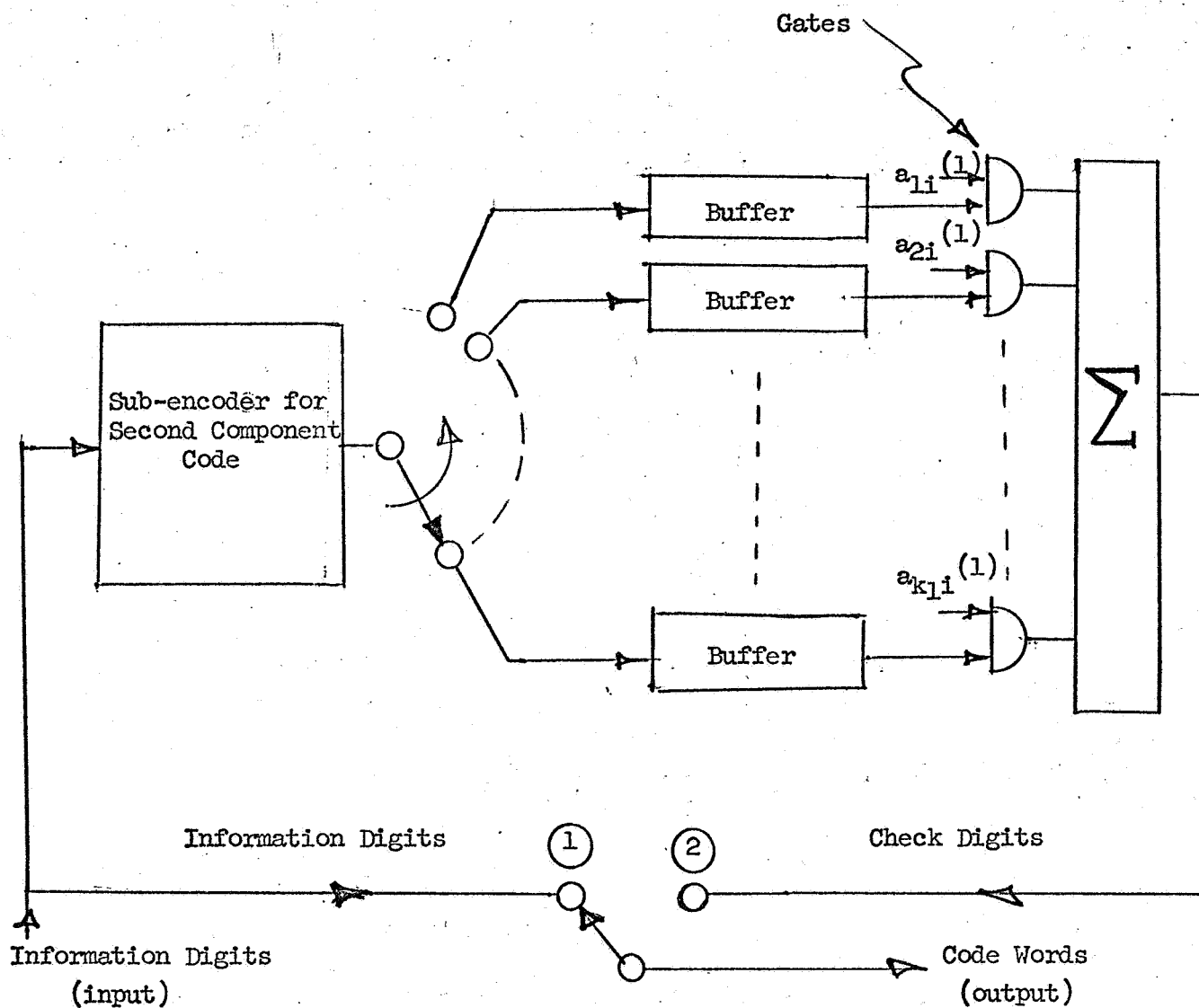
$$\underbrace{a_{i1}^{(1)} \quad a_j^{(2)} \quad a_{i2}^{(1)} \quad a_j^{(2)} \quad a_{i3}^{(1)} \quad a_j^{(2)} \quad \dots \quad a_{in_1}^{(1)} \quad a_j^{(2)}}_{}, \quad (2)$$

where  $a_j^{(2)}$  is a row vector of  $G_2$ . A code word of the iterated code is a linear combination of these vectors. Hence the encoder for the iterated code follows as shown in Fig. 3. The information in groups of  $k_2$  digits each is fed sequentially into the sub-encoder for the second component code.

The sub-encoder may be designed according to either Fig. 1 or Fig. 2 as desired. However, one configuration will always require fewer delay elements and modulo two adders. Every subblock of  $n_2$  digits of the output of the sub-encoder is fed into a buffer storage, for a total of  $k_1$  subblocks. The gates are controlled by the elements in the columns of the first component code generator matrix

$(a_{1i}^{(1)} \ a_{2i}^{(1)} \ \dots \ a_{k_1 i}^{(1)})^T$  and the order of  $i$  is from right to left, i.e.,

$n_1, n_1-1, n_1-2, \dots, 1$ . Each value of  $i$  controls the gating of a group of  $n_2$  digits from the buffer storage. The outputs of the gates are summed, modulo two, to form the checking digits.



Switch S is in position 1 for incoming information digits, and then is moved to position 2 for the checking digits.

Fig. 3 Encoder for an Iterated Code

The controlling elements  $a_{11}^{(1)}, \dots, a_{k_1 1}^{(1)}$  can be easily generated by a sequence generator connected according to  $g_1(x)$ , the generator polynomial of the first component code. Also, it should be noted that the encoder of Fig. 3 can be operated in four different modes. The input/output connections and timing of the sub-encoder can be changed (by employing suitable switching) to those of its dual. Alternatively, the controlling element generator can be altered to generate sequences according to  $h_1(x)$ , the recursive polynomial of the first component code.

### Error-Locating Codes

Let the parity-check matrix of an error-locating code be the product of those of two component codes,  $H_1$  and  $H_2$ :

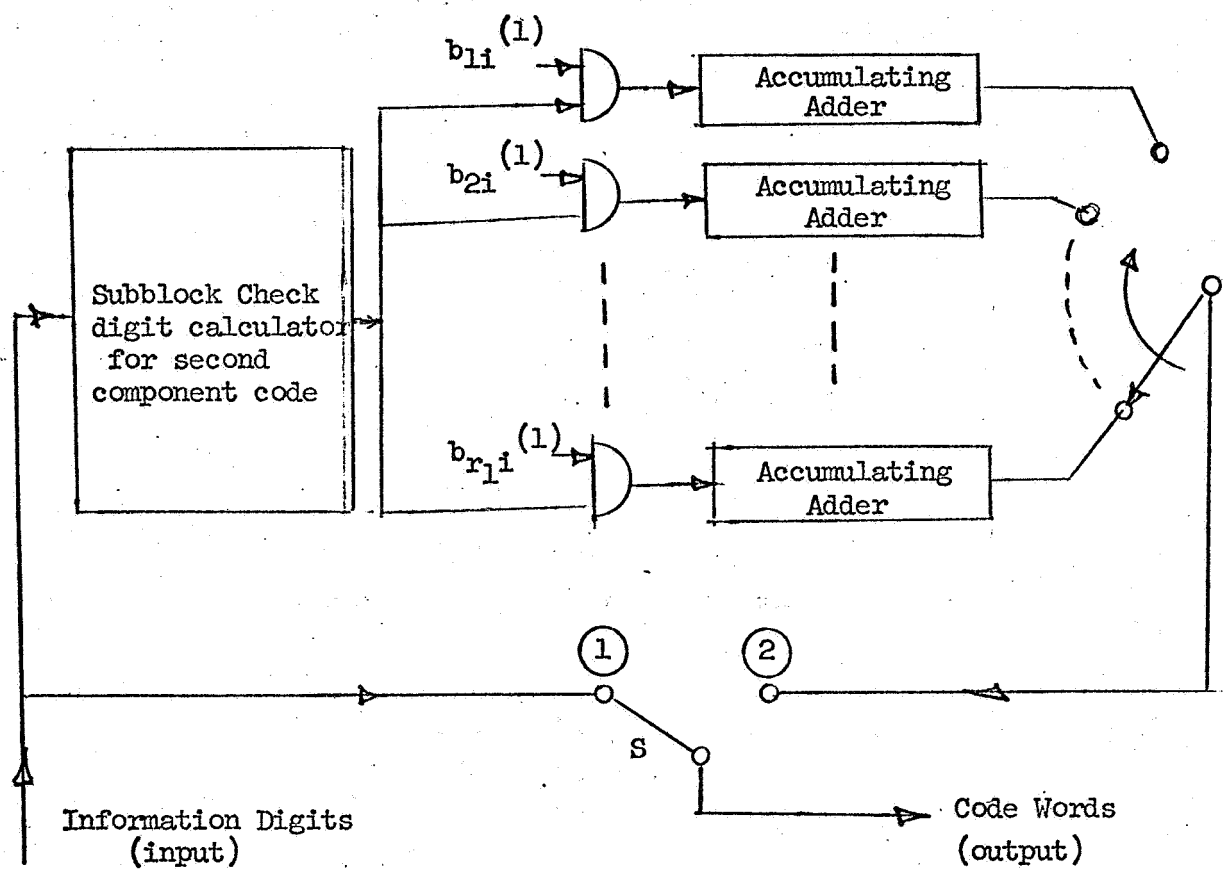
$$H = H_1 \otimes H_2 = \begin{bmatrix} b_{11}^{(1)} & b_{12}^{(1)} & \dots & b_{1n_1}^{(1)} \\ b_{21}^{(1)} & b_{22}^{(1)} & \dots & b_{2n_1}^{(1)} \\ \vdots & \vdots & & \vdots \\ b_{r_1 1}^{(1)} & b_{r_1 2}^{(1)} & \dots & b_{r_1 n_1}^{(1)} \end{bmatrix} \otimes \begin{bmatrix} b_{11}^{(2)} & b_{12}^{(2)} & \dots & b_{1n_2}^{(2)} \\ b_{21}^{(2)} & b_{22}^{(2)} & \dots & b_{2n_2}^{(2)} \\ \vdots & \vdots & & \vdots \\ b_{r_1 1}^{(2)} & b_{r_1 2}^{(2)} & \dots & b_{r_1 n_2}^{(2)} \end{bmatrix}, \quad (3)$$

where  $r_1 = n_1 - k_1$  and  $r_2 = n_2 - k_2$ . Then a code word of the error-locating code satisfies all the parity-check equations in  $H$ , obtainable from the row vectors of

$H$  which are of the form  $b_{i1}^{(1)} H_2 \dots b_{in_1}^{(1)} H_2$ . This encoding scheme may be implemented by subdividing the information digits (together with zeros in the check digit positions) into  $n_1$  subblocks of  $n_2$  digits each and feeding each group into a check digit calculator connected according to the second component code. The resultant check digits of each subblock are gated by the sequence

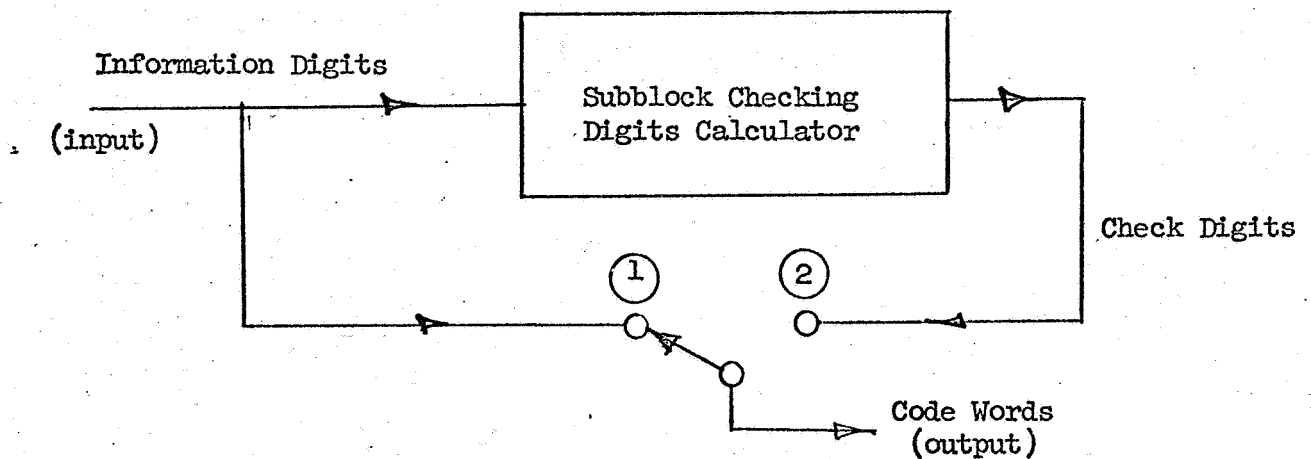
$b_{i1}^{(1)} \dots b_{in_1}^{(1)}$  and summed subblock-by-subblock to form the check digits of the code word. This encoder is shown in Fig. 4.

The parity-check matrix of the first component code controls the gating and thus the sequence may be generated by a shift register connected according to  $h_1(x)$ . It can be noted that the subblock check digit calculator can be converted into a sub-encoder (like that of Fig. 3) as shown in Fig. 5.



Switch S is at position 1 for information digits and 2 for check digits

Fig. 4 Encoder for Error-Locating Codes



Switch S is at 1 for information digits incoming and at 2 for check digits.

Fig. 5 Check Digit Calculation and Encoder Conversion

By changing the second component code to its dual there results two modes. Moreover, the first component code which generates the controlling sequency may also be changed to its dual. In this way four modes are achieved with the scheme of Fig. 4.

#### Eight-Mode Encoder

An eight mode encoder is then achieved by combining the encoders of Figs. 3 and 4. The combination poses no difficulty because the sub-encoder may be converted to a sub-check-digit-calculator by altering the input/output connections as shown in Fig. 5 and the accumulating adder is actually the buffer storage with a single feedback loop. Hence by the addition of judiciously placed gates or switches we may alter the encoder to perform any one of eight encoding functions, consisting of four iterated codes and four error-locating codes.

For linear codes a syndrome<sup>8</sup> calculator may be obtained by a slight modification of the encoder, but the decoding procedure from this point on may require no additional circuits or up to eight separate circuits depending upon the decoding strategy to be used.

An example of this encoder is the encoder formed on the basis of a B-C-H (15,5) three error-correcting code (with generator matrix  $G_1$  and parity-check matrix  $H_1$ ) and the Hamming (7,4) single error-correcting code (with generator matrix  $G_2$



and parity-check matrix  $H_2$ ). The eight resultant codes and their capabilities are given in Table 1. An eight mode encoder/decoder has been designed and is being constructed using the ideas presented here.

An error analysis using the encoder in a binary symmetric channel has also been performed which indicates how the encoder should be used with and without feedback channels.

In summation, the important advantages of this encoder are:

- (1) The encoder may be easily switched through any one of eight modes, four of these being iterated codes (typically high redundancy codes) and four being the error-locating codes (typically low redundancy).
- (2) The code word block length remains constant through a wide range of code redundancies for ease of synchronization and possibly decoding.
- (3) The encoder may be used to form the basis for an adaptive coding system, i.e., low redundancy codes may be used in low signal-to-noise environments, and conversely.
- (4) The encoder scheme is optimal in the sense that no additional clock periods are required to generate a code word, e.g., a (105,16) tensor product code requires 105 clock periods and no more.
- (5) The encoder lends itself to various feedback strategies, especially the error-locating codes which detect the erroneous transmitted subblocks.

Table 1

Code	Generator Matrix	Parity-Check Matrix	Min Distance	Functions
(15,5)	$G_1$	$H_1$	7	3-error-correcting
(15,10)	$H_1$	$G_1$	4	1-error-correcting 2-error-detecting
(7,4)	$G_2$	$H_2$	3	1-error-correcting
(7,3)*	$G_2'$	$H_2'$	4	1-error-correcting 2-error-detecting
(105,20)	$G_1 \otimes G_2$		21	10-error-correcting
(105,15)	$G_1 \otimes G_2'$		28	13-error-correcting 14-error-detecting
(105,40)	$H_1 \otimes G_2$		12	5-error-correcting 6-error-detecting
(105,30)	$H_1 \otimes G_2'$		16	7-error-correcting 8-error-detecting
(105,75)		$H_1 \otimes H_2$	3	$(3,2)^{**}$
(105,65)		$H_1 \otimes H_2'$	4	$(3,3)$
(105,90)		$G_1 \otimes H_2$	3	$(2a,2)^{***}$
(105,85)		$G_1 \otimes H_2'$	4	$(2a,3)$

\*This code is obtained by interchanging the generating polynomial and recursive polynomial of the (7,4) code.

\*\* $(s,t)$  denotes the code is capable of locating  $s$  erroneous subblocks with no more than  $t$  errors in each.

\*\*\* $(2a,t)$  denotes the code is capable of locating 2 adjacent erroneous subblocks with no more than  $t$  errors in each.

### References

1. P. Elias, "Error-Free Coding", IRE Transactions PGIT-4, 1954, pp. 29-37.
2. D. Slepian, "Some Further Theory of Group Codes", Bell System Technical Journal, 39, 1219-1252, 1960.
3. J. K. Wolf and E. Elspas, "Error-Locating Codes -- A New Concept in Error Control", IEEE Transactions on Information Theory, Vol. IT-9, No. 2, (April 1963), pp. 113-117.
4. S. H. Chang and L. J. Weng, "Error-Locating Codes", 1965 IEEE International Convention Record, Part 7, pp. 252-258.
5. J. K. Wolf, "On Codes Derivable from the Tensor Product of Check Matrices", IEEE Transactions on Information Theory, Vol. IT-11, No. 2, (April 1965), pp. 281-284.
6. J. K. Wolf, "On an Extended Class of Error-Locating Codes", Information and Control, Vol. 8, No. 2, (April 1965), pp. 163-169.
7. D. T. Tang, "Dual Codes as Variable Redundancy Codes", 1965 IEEE International Convention Record, Part 7, pp. 220-226.
8. W. W. Peterson, Error-Correcting Codes, MIT Press, Cambridge, Massachusetts, 1961.
9. L. J. Weng, "On Linear Product Codes and Their Duals", Scientific Report No. 4, Northeastern University, Boston, Massachusetts, (August 1966).